

Practical algorithms^{*}: Computing β_N

Cheng Guan Koay[†]

April 19, 2008

In this note, we will present a simple algorithm for computing β_N , which was used in [1]. Briefly, β_N is given by:

$$\beta_N = \sqrt{\pi/2} \frac{(2N-1)!!}{2^{N-1}(N-1)!}, \quad (1)$$

where N is a positive integer, the double factorial of N is given by $N!! = N \times (N-2) \cdots$ with $0!! \equiv 1$ while the factorial of N is given by $N! = N \times (N-1) \cdots 2 \times 1$ with $0! \equiv 1$.

The expression in Eq.(1) is elegant and compact but it is not practical for numerical computation in a finite precision system that does not have special routines or packages such as **BigDecimal** and **BigInteger**s in JavaTM[2] because it is expressed as a ratio of two big numbers—the factorials. For example, $(2N-1)!!$ for $N=64$ is 1.647×10^{107} .

Fortunately, this expression can be further simplified to ¹

$$\beta_N = \sqrt{\frac{\pi}{2}} \prod_{k=2}^N \frac{k-1/2}{k-1}. \quad (2)$$

The derivation of the above expression is shown in the appendix. Note that the formula in Eq.(2) is now expressed as a product of many small ratios. For completeness, the Java code for computing β_N is shown below:

```
public static double beta(int N){  
  
    double p = 1.0;  
    for(int i=2; i<=N; i++){  
        p *= ((i-0.5)/(i-1.0));  
    }  
  
    return Math.sqrt(0.5*Math.PI)*p;  
}
```

^{*}The main reason for writing up these notes is that I often have to spend some time rediscovering what I have found long ago about certain not-so-obvious (nontrivial) techniques for computing certain quantities if I do not have a record of how these techniques were invented or devised. Frankly, these notes are very helpful to me so that I can remind myself the main reasoning behind the techniques. As R.W. Hamming once put it, “The purpose of computing is insight, not numbers”, my goal here is to present practical algorithms that I have designed and used in my research work in an intelligible manner so that interested readers may be able understand and use them in their own work. Of course, comments from readers are welcome.

[†]Contact info: guankoac@mail.nih.gov

¹The expression in Eq.(2) can also expressed in term of Gamma function. That is, $\beta_N = \sqrt{2}\Gamma(n+1/2)/\Gamma(n)$. Again, this formula is still expressed as a ratio of two large number.

Appendix

$$\beta_N = \sqrt{\frac{\pi}{2}} \frac{(2N-1)!!}{2^{N-1}(N-1)!} \quad (3)$$

$$= \sqrt{\frac{\pi}{2}} \frac{(2N-1) \times (2N-3) \times (2N-5) \times \cdots \times 3 \times 1}{2^{N-1}(N-1) \times (N-2) \times (N-3) \times \cdots \times 2 \times 1} \quad (4)$$

$$= \sqrt{\frac{\pi}{2}} \frac{(2N-1)/2 \times (2N-3)/2 \times (2N-5)/2 \times \cdots \times 3/2 \times 1}{(N-1) \times (N-2) \times \cdots \times (2) \times (1)} \quad (5)$$

$$= \sqrt{\frac{\pi}{2}} \frac{(2(N)-1)/2 \times (2(N-1)-1)/2 \times (2(N-2)-1)/2 \cdots}{(N-1) \times (N-2) \times (N-3) \cdots \times (2) \times (1)} \quad (6)$$

$$= \sqrt{\frac{\pi}{2}} \frac{((N)-1/2) \times ((N-1)-1/2) \times ((N-2)-1/2) \cdots}{((N)-1) \times ((N-1)-1) \times ((N-2)-1) \times \cdots} \quad (7)$$

$$= \sqrt{\frac{\pi}{2}} \frac{(N-1/2)}{(N-1)} \times \frac{((N-1)-1/2)}{((N-1)-1)} \times \frac{((N-2)-1/2)}{((N-2)-1)} \times \cdots \quad (8)$$

$$= \sqrt{\frac{\pi}{2}} \prod_{k=2}^N \frac{k-1/2}{k-1}. \quad (9)$$

Note that there are $N-1$ factors in $(2N-1)!!$, not counting the factor 1, and that every factor is an odd number for any positive integer N . By dividing each factor by 2 in a descending order starting from the largest factor, we arrive at Eq.(5) with $3/2$ being the last factor after the division, again not counting the factor 1. This is the main reason why k in Eq.(9) starts at 2.

References

- [1] Koay, C.G. and Bassar, P.J. *Analytically exact correction scheme for signal extraction from noisy magnitude MR signals*. J. Magn. Reson. **179**, 317-322 (2006).
- [2] Arnold, K., Gosling, J. and Holmes, D. *The JavaTM programming language*. 4th Ed. Prentice Hall; New York: 2005.